ASCII
Caesar Cipher

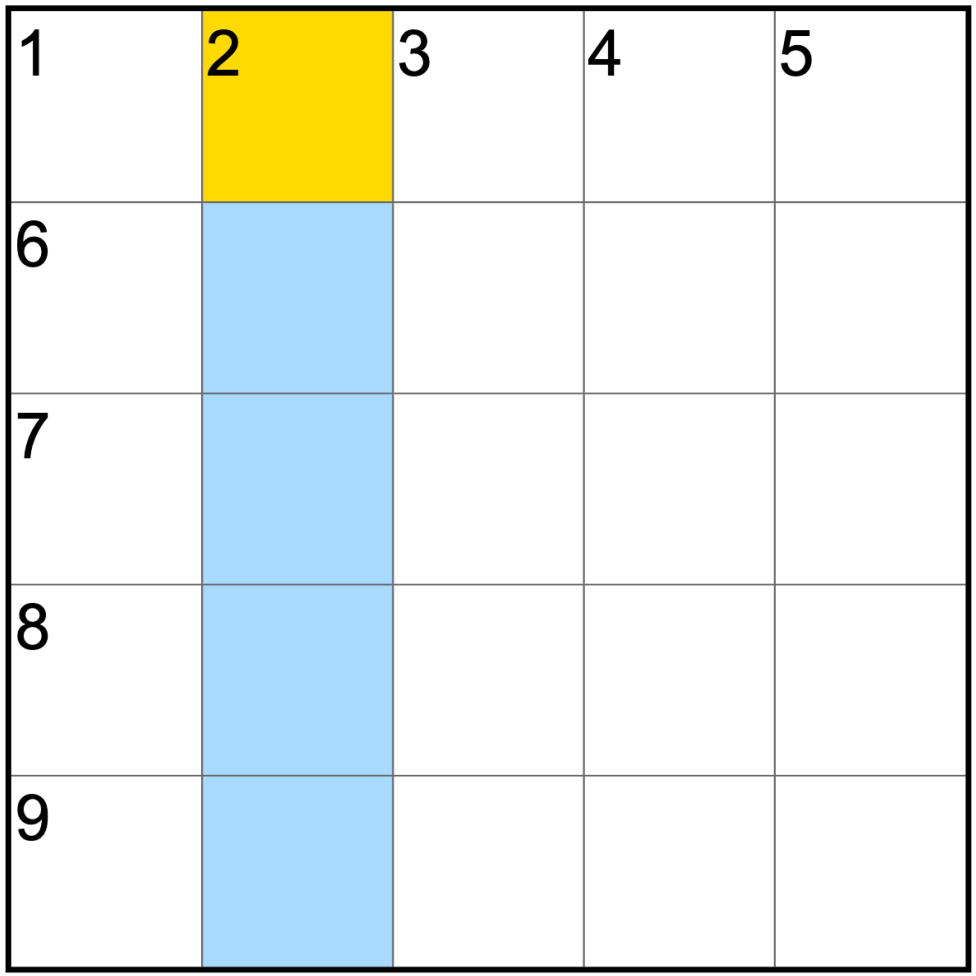
Aaron Stevens (azs@bu.edu)

Boston University Department of Computer Science

BOSTON UNIVERSITY

NY Times Mini Crossword 2/19/20

2D Computer character code



ACROSS

1 Elbows on the dinner table?

6 Colorado ski resort that hosts the Winter X Games

7 Negative R.S.V.P.

8 Things actors memorize

9 Minutes in an hour

DOWN

1 Toys used for making sand castles

2 Computer character code

3 Big name in leggings

4 Core belief

5 Fidgeting in one's seat

Converting Between Characters and Numbers

ASCII values

Converting Between Characters and Numbers

ASCII values

```
abcdefghijklmnopqrstuvwxyz
97  99  101 103 105 107 109 111 113 115 117 119 122

ABCDEFGHIJKLMNOPQRSTUVWXYZ
65  67  69  71  73  75  77  79  81  83  85  87  90
```

Conversion functions

`ord(c)`

input: a one-character string, `c`
returns: an integer, the ASCII value of `c`

Examples

```
>>> ord('e')
101
>>> ord('G')
71
```

Converting Between Characters and Numbers

ASCII values

```
abcdefghijklmnopqrstuvwxyz
97  99  101 103 105 107 109 111 113 115 117 119 122

ABCDEFGHIJKLMNOPQRSTUVWXYZ
65  67  69  71  73  75  77  79  81  83  85  87  90
```

Conversion functions

`ord(c)`

input: a one-character string, `c`
returns: an integer, the ASCII value of `c`

`chr(n)`

input: an integer ASCII value
returns: the one-character string for that ASCII value

Examples

```
>>> ord('e')
101
>>> ord('G')
71
```

```
>>> chr(101)
'e'
>>> chr(71)
'G'
```

Encryption

original message

```
'my password is foobar'
```

Encryption

original message *encrypted message*

`'my password is foobar'` → `'pb sdvvzrug lv irredu'`

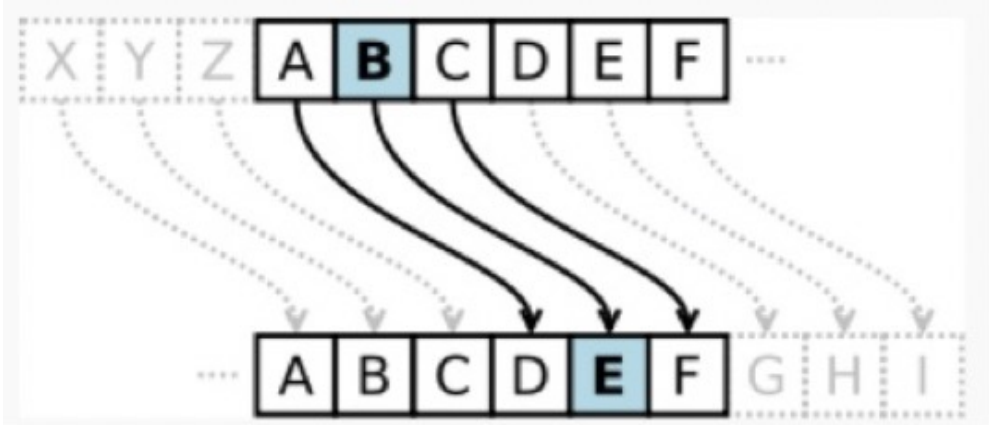
Problem 12: The `cipher(s, n)` function

- You will write a `cipher` function, to cipher a string `s`:

```
>>> cipher('hello!', 1)
'ifmmp!'
>>> cipher('hello!', 2)
'jgnnq!'
>>> cipher('hello!', 4)
'lipps!'
```

Caesar Cipher Encryption

- Each letter is shifted ("rotated") forward by some number of places.
- Example: a shift of 3
  `'b' → 'e'`



Example: Caesar Cipher with a Shift/Rotation of 13

- `'a'` → `'n'`
  `'b'` → `'o'`
  `'c'` → `'p'`
  etc.

`'n'` → `'a'`
`'o'` → `'b'`
`'p'` → `'c'`

Example: Caesar Cipher with a Shift/Rotation of 13

- `'a'` → `'n'`
  `'b'` → `'o'`
  `'c'` → `'p'`
  etc.

  `'n'` → `'a'`
  `'o'` → `'b'`
  `'p'` → `'c'`

- Using `chr()` and `ord()`:

```
>>> chr(ord('a') + 13)
'n'
```

Example: Caesar Cipher with a Shift/Rotation of 13

- `'a'` → `'n'`
  `'b'` → `'o'`
  `'c'` → `'p'`
  etc.

  `'n'` → `'a'`
  `'o'` → `'b'`
  `'p'` → `'c'`

- Using `chr()` and `ord()`:

```
>>> chr(ord('a') + 13)          # 97 + 13 = 110
'n'
```

Example: Caesar Cipher with a Shift/Rotation of 13

- `'a'` → `'n'`
  `'b'` → `'o'`
  `'c'` → `'p'`
  etc.

  `'n'` → `'a'`
  `'o'` → `'b'`
  `'p'` → `'c'`

- Using `chr()` and `ord()`:

```
>>> chr(ord('a') + 13)          # 97 + 13 = 110
'n'
>>> chr(ord('P') + 13)
']'
```

Example: Caesar Cipher with a Shift/Rotation of 13

- `'a'` → `'n'`
  `'b'` → `'o'`
  `'c'` → `'p'`
  etc.

  `'n'` → `'a'`
  `'o'` → `'b'`
  `'p'` → `'c'`

- Using `chr()` and `ord()`:

```
>>> chr(ord('a') + 13)          # 97 + 13 = 110
'n'
>>> chr(ord('P') + 13)          # 80 + 13 = 93
']'
```

Example: Caesar Cipher with a Shift/Rotation of 13

- `'a'` → `'n'`
  `'b'` → `'o'`
  `'c'` → `'p'`
  etc.

  `'n'` → `'a'`
  `'o'` → `'b'`
  `'p'` → `'c'`

- Using `chr()` and `ord()`:

```
>>> chr(ord('a') + 13)          # 97 + 13 = 110
'n'
>>> chr(ord('P') + 13)          # 80 + 13 = 93
']'                             # wrap-around?
```

Example: Caesar Cipher with a Shift/Rotation of 13

- `'a' → 'n'`
  `'b' → 'o'`
  `'c' → 'p'`
  etc.

  `'n' → 'a'`
  `'o' → 'b'`
  `'p' → 'c'`

- Using `chr()` and `ord()`:

```
>>> chr(ord('a') + 13)          # 97 + 13 = 110
'n'
>>> chr(ord('P') + 13 - 26)     # 80 + 13 - 26 = 67
'C'                             # wrap-around!
```

Note: about the wrap-around:

Solve this problem for lower-case letters first! Go back and solve it for upper-case letters later.

- We can use the following to determine if character `c` is lower-case:

  ```
  if 'a' <= c <= 'z':
  ```

  → maximum valid lower-case letter is `'z'`

  → minimum valid lower-case letter is `'a'`

- We can use the following to determine if character `c` is upper-case:

  ```
  if 'A' <= c <= 'Z':
  ```

  → maximum valid upper-case letter is `'Z'`

  → minimum valid upper-case letter is `'A'`